

## Algorithmentheorie

### Übungsblatt 1 (Abgabe am 30.10.2023, 12:15 Uhr)

#### Übung 1.1

(7 Punkte)

Ihr arbeitet als Analysten in einer Investmentfirma. Mit Hilfe von Simulationen seid Ihr in der Lage, die Preisentwicklung der Aktie eines Unternehmens für die nächsten  $n$  Tage zu berechnen. Ihr habt also Daten zur Verfügung, welche beispielhaft die folgende Form haben:

Tag	0	1	2	3	4	5	6	7	...
Änderung (in \$)	3	7	-4	-8	10	-2	4	-3	...

Das Array enthält einen Eintrag der für jeden Tag den Gewinn bzw. Verlust zur Zeit des Börsenschlusses des zugehörigen Tages angibt. Wir nehmen an, dass das Array nicht nur negative Zahlen enthält. Eure Aufgabe ist es, auf Basis dieser Daten, genau einen der nächsten  $n$  Tage auszuwählen, um morgens die Aktie des Unternehmens zu kaufen. Außerdem sollt Ihr genau einen dieser  $n$  Tage auswählen, um die Aktie am Abend wieder zu verkaufen. Dabei sollt Ihr einen möglichst hohen Profit erzielen. Euer Ziel ist es also, zwei Indizes  $i \leq j$  mit dem bestmöglichen Profit zu finden, d.h., Ihr sucht  $i$  und  $j$ , so dass  $\sum_{k=i}^j A[k]$  maximiert wird.

Gebt den Pseudocode eines Algorithmus an, der das Problem löst. Argumentiert kurz, warum Euer Algorithmus korrekt ist und welche Laufzeit dieser erzielt.

#### Übung 1.2

(7 Punkte)

Gegeben sei ein sortiertes  $n$ -elementiges Array  $A = [a_0, a_1, \dots, a_{n-1}]$  von natürlichen Zahlen, sowie eine natürliche Zahl  $k$ . Gesucht ist ein Indexpaar, dessen Elemente summiert  $k$  ergeben. Formal: Gib ein Indexpaar  $(i, j)$  aus, sodass  $0 \leq i < j \leq n - 1$  und  $a_i + a_j = k$ . Falls kein solches Indexpaar existiert, gib  $-1$  aus.

Gebt den Pseudocode eines Algorithmus an, der das Problem mit linearer Worst-Case Laufzeit löst, d.h., die Worst-Case Laufzeit ist  $an + b$  für geeignete Konstanten  $a, b$ . Argumentiert, warum Euer Algorithmus korrekt ist und tatsächlich die geforderte Laufzeit erzielt. Was ist die Best-Case Laufzeit?

#### Übung 1.3

(6 Punkte)

Gegeben sei ein  $n$ -elementiges Array  $A = [a_0, a_1, \dots, a_{n-1}]$  von natürlichen Zahlen. Wir möchten analysieren wie viele Vergleiche (siehe Zeile 4 im Pseudocode aus der Vorlesung) Insertion Sort benötigt um  $A$  zu sortieren:

- (a) Wie viele Vergleiche führt Insertion Sort *mindestens* aus um  $A$  zu sortieren? Begründe warum deine Antwort korrekt ist und beschreibe wie das Eingabearray aussehen muss, damit dieser Best Case erreicht wird.

- (b) Wie viele Vergleiche führt Insertion Sort *maximal* aus um A zu sortieren? Begründe warum deine Antwort korrekt ist und beschreibe wie das Eingabearray aussehen muss, damit dieser Worst Case erreicht wird.